Chapter 5
Simulations and Results

5.1 MATLAB® as a PC-Based Simulation Software

To simulate the motion of a multi-unit tracked vehicle such as the CHS on a digital computer, the equations of motion of the system that govern and describe its motion have to be implemented on the digital computer by using simulation software. The simulation software available nowadays mainly consists of programming languages, simulation languages, and simulation packages [Kheir, 1996]. For programming languages, C, C++, and PASCAL are widely accepted and used due to their execution speed and programming flexibility; however, the graphics presentation for either animating or plotting the result can be tedious, if one wants to do it with these programming languages. Thus, these programming languages do not suit our requirement. On the other hand, the simulation packages in mechanical applications such as ADAMS and DADS do offer these capabilities; nevertheless, the simulation cost can be expensive. It is also difficult to customize these simulation packages to fit our needs since these packages do not contain a toolbox for track-terrain interaction model, which is our major interest. Even otherwise, the price of the toolbox would be costly.

As a result, we narrow our choices for the simulation languages that are interpreter oriented, similar to programming languages, but are specifically used for simulation applications. There are several simulation languages available on the market, for example, CSSL, CSMP, ACSL, and etc. MATLAB® is one of the most popular simulation languages used by many scientists and researchers worldwide. Although the primary intention of developing MATLAB® was to be used in control engineering applications, the capabilities of MATLAB® in matrix manipulation and 2-D graphics presentation fit perfectly with our needs. Furthermore, a simulation program written by using MATLAB® can be run on a personal computer equipped with a 200 MHz microprocessor or above with an acceptable speed, and this can recover the cost involved in performing simulation. Accordingly, MATLAB® is selected to be the simulation

software in this research, and the personal computer is chosen to be the simulation platform.

5.2 Numeric Integration of a System of ODEs

The numerical methods for solving a system of ordinary differential equations can be categorized into two methods. One is called one-step methods, and the other is called multi-step methods. Each method is subdivided into two types, constant and adaptive step sizes [Chapra and Canale, 1988]. Among these numeric integration methods, a fourth-order Runge-Kutta method is well known and widely used in many engineering applications since it is simple to program and has high accuracy and reliability. The comparison of several numeric integration methods is shown in Table 5.1. The fourth-order Runge-Kutta method is a single-step method with constant step size. The advantages that the one-step methods have over the multi-step methods are that all single-step methods are self-starting and that they require less computational effort and resources. Moreover, with constant step size, we can accurately estimate the total elapsed time taken for each simulation, and ensure that the numeric integration routine is always stable, which is not always true for the adaptive step size methods.

The forth-order Runge-Kutta method, which provides a numerical solution to $\dot{x} = f(x)$ for x with $x(0) = x_0$ and a constant step size (time step) Δt, can be stated as follows:

$$x_{t+1} = x_t + \frac{1}{6}(z_1 + 2z_2 + 2z_3 + z_4) \tag{5.1}$$

where x_t = the state vector at time t

$$z_1 = f(x_t)\Delta t$$

$$z_2 = f(x_t + \frac{z_1}{2})\Delta t$$

$$z_3 = f(x_t + \frac{z_2}{2})\Delta t$$

$$z_4 = f(x_t + z_3)\Delta t$$

Method	Starting Values	Iterations Required	Global Error	Ease of Changing Step Size	Programming Effort	Comments
One step						
Euler's	1	No	$O(h)$	Easy	Easy	Good for quick estimates
Heun's	1	Yes	$O(h^2)$	Easy	Moderate	—
Improved polygon	1	No	$O(h^2)$	Easy	Moderate	—
Second-order Ralston	1	No	$O(h^2)$	Easy	Moderate	The second-order RK method that minimizes truncation error
Fourth-order RK	1	No	$O(h^4)$	Easy	Moderate	Widely used
Adaptive fourth-order RK or RK-Fehlberg	1	No	$O(h^5)$*	Easy	Moderate to difficult†	Error estimate allows step-size adjustment
Multistep						
Non-self-starting Heun	2	Yes	$O(h^3)$*	Difficult	Moderate to difficult†	Simple multistep method
Milne's	4	Yes	$O(h^5)$*	Difficult	Moderate to difficult†	Sometimes unstable
Fourth-order Adams	4	Yes	$O(h^5)$*	Difficult	Moderate to difficult†	

*Provided error estimate is used to modify the solution.
†With variable step size.

Table 5.1. Comparison of alternative methods for solving a system of ODEs

From Chapter 4, Eq. 4.50 can be rewritten in the following forms:

$$[A(s)]\{\dot{s}\} = \{y(s)\} \tag{5.2}$$

where

$$\{\dot{s}\} = \{x\}$$

Thus, the values of z_1 in Eq. 5.1 are determined by substituting the values at time t of the state vector, $\{s\}$, which comprises the positions and velocities of each component in the CHS, into Eq. 5.2 and then solving for the acceleration and internal force vector, $\{\dot{s}\}$. Only the values of the accelerations are extracted and used for calculating z_1. The process is repeated until all values of z_2, z_3, and z_4 are computed. After Eq. 5.1 has been evaluated, we obtain the velocity part of the state vector at time $t+1$. However, in order to compute the position part of the state vector, we utilize a much simpler numeric integration method, called Heun's Method. In this case, we assume that the velocities are linearly varying over the time step, Δt, when the time step is small enough. Accordingly, one can obtain the position part of the state vector at time $t+1$ by multiplying the average

value of the velocity vectors at time t and t+1 by the time step and adding it to the position vector at time t, as written in the following form.

$$\{p_{t+1}\} = \{p_t\} + \left\{\frac{v_t + v_{t+1}}{2}\right\}\Delta t \tag{5.2}$$

5.3 The Simulation of an MBC Maneuvered on Flat Ground

The study of the turning motion of tracked vehicles has been investigated in many aspects for a long time. Most of the objectives of the study are related to designing the vehicles themselves to yield the maximum performance and meet the requirements for each mission. Only a few of these works deal with how to control the tracked vehicles. Before one can begin to design a tracked vehicle controller, a high fidelity dynamic model of the tracked vehicle must be known. In this research, the existing models of the tracked vehicles are first investigated and then adapted to fit our needs. The modified tracked vehicle model is then programmed and numerically solved on a digital computer. Later, the model validation will be done by means of comparing the simulation result with the experimental data from a field test. Since the modeling process has been already discussed in the previous two chapters, the simulation result and the model validation will be presented in this section.

The Simulation Result of an MBC Turning at Constant Speed:

A MATLAB® code presented in Appendix A is a simulation program for simulating the trajectory of an MBC subject to specified track velocities. The inputs of the program consist of the parameters of the MBC, terrain, numeric integrator, and velocity commands. Hence, one can generate many simulation results based on different inputs, and use the results to study the effect of the parameter of interest to the motion of the MBC. The outcomes of the program are the MBC's positions and velocities, track slippages, the amount of shift of the instantaneous center, and the radius of turning.

The following table contains the list of parameters input to the simulation program.

MBC	
W_M	75000 lb
Ig	$1.3373*10^5$ lb*ft*s^2
W,mbc	8 ft
L,mbc	25 ft
B	7.17 ft
l	6 ft
w	1.25 ft
e_x	0 ft
e_y	1 ft
Vr	1.46 ft/s
Vl	0.65 ft/s
Terrain	
c	0 lbf/ft^2
ϕ	0.75 rad
K	0.033 ft
μ_x	0.1
μ_y	0.5

Table 5.2. MBC and Terrain Parameters

The total simulation time and time step are set to be 30 and 0.005 seconds, respectively. The simulation results are shown in the figures on next page.

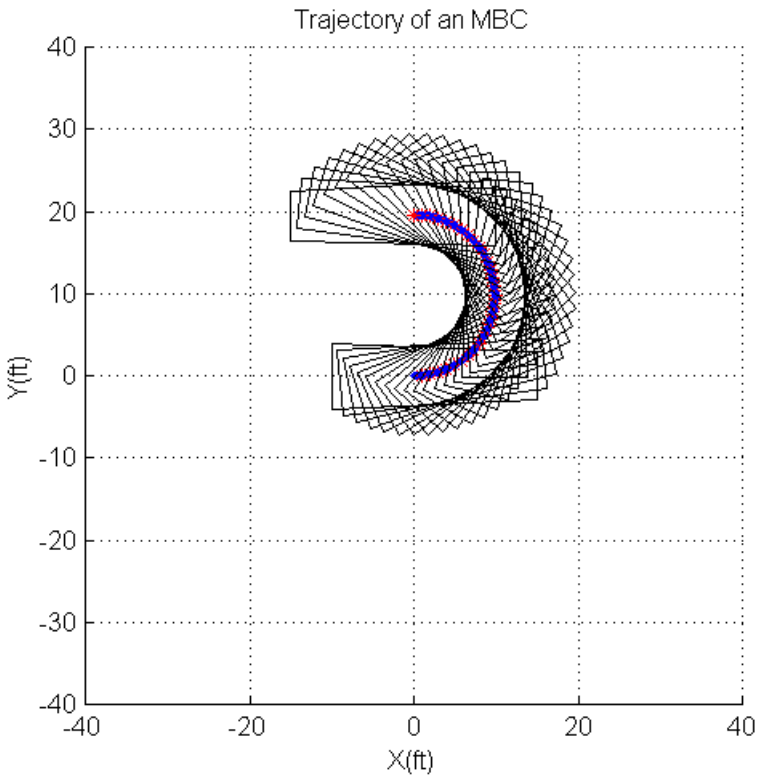

Figure 5.1. Trajectory of an MBC

The trajectory of an MBC turning at constant track speeds is shown in the figure above. Both right and left track speeds ramp up with constant accelerations until they reach the specified speeds within 0.1 second, which is close to what actually happened in the field test. At the beginning of the simulation, the geometric center of the MBC coincides with the origin of the global coordinate system, and the MBC heading direction points toward left. In Fig. 5.1, the curved line represents the trajectory of motion of the geometric center of the MBC, and the asterisks (*) are used as simulation time marks stamped for every second elapsed. The figure clearly shows that the MBC makes a 180-degree turn with a constant radius of turning within approximately 29.5 seconds.

Figs. 5.2, 5.3, and 5.4 show the profile of linear velocities and angular velocity of the MBC during the motion in the body-fixed coordinate system. From these Figs., we obtain the values of the MBC velocities at the steady state as follows:

$$v_x = 1.05 \quad \text{ft/s}$$

$$v_y = -0.002 \quad \text{ft/s}$$

$$\omega = 0.105 \quad \text{rad/s}$$

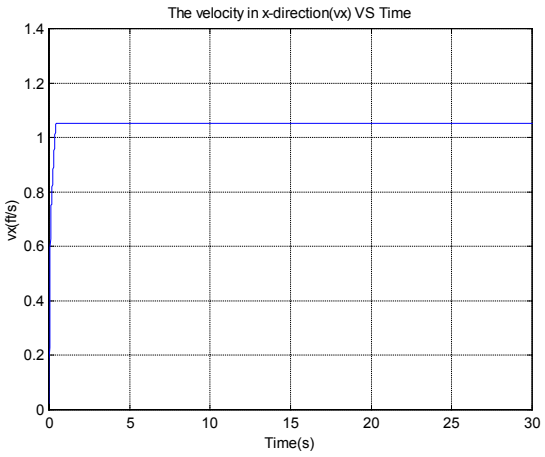

Figure 5.2. v_x VS Time

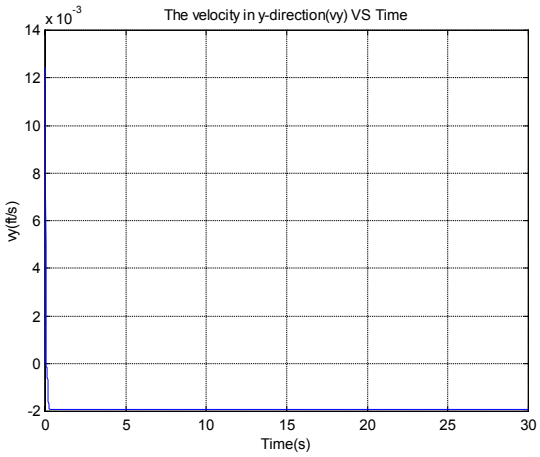

Figure 5.3. v_y VS Time

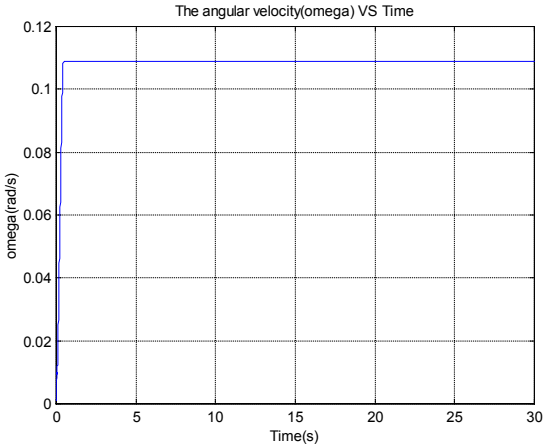

Figure 5.4. ω VS Time

The time histories of the slips occurring beneath both right and left tracks are shown in Fig. 5.5 and 5.6, respectively.

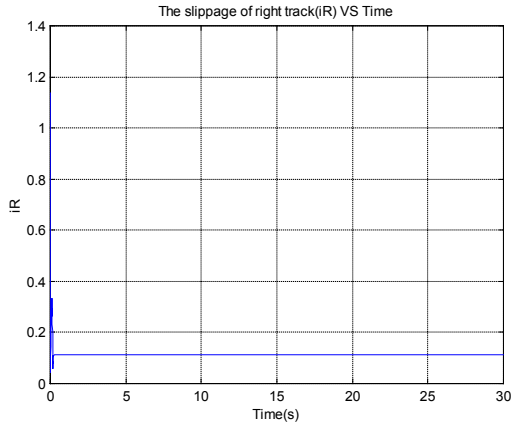

Figure 5.5. *iR* VS Time

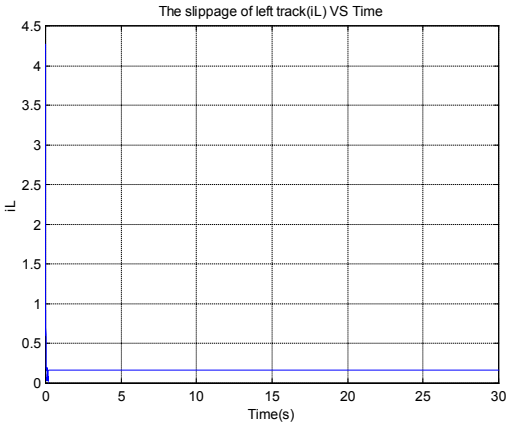

Figure 5.6. *iL* VS Time

The values of the slippage of right and left tracks at the steady state are:

$$iR = 0.11$$

$$iL = 0.16$$

Figs. 5.7 and 5.8 shows the plot of the amount of shift of the instantaneous center (d) and the radius of turning of the MBC (Rt) over time. At the steady state, the values of d and Rt are 0.018 ft and 10 ft, respectively.

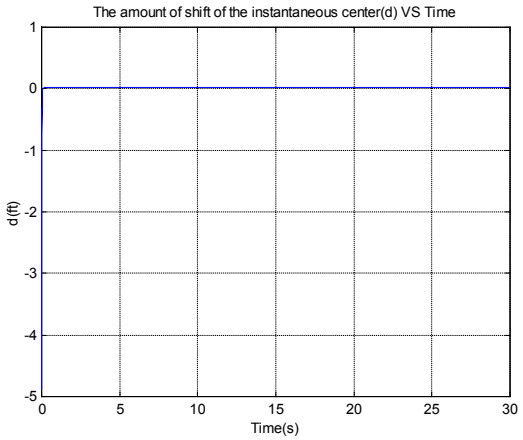

Figure 5.7. d VS Time

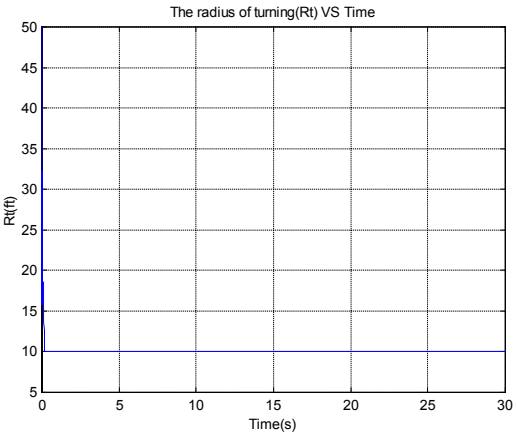

Figure 5.8. Rt VS Time

The Field Test:

To validate the model, a field test was conducted, and the results from the test could be used to verify how accurate the model is in predicting the motion of the real MBC. The MBC used in the experiment was an MBC equipped with a feeder breaker. The MBC parameters are the same the ones that are into the simulation program, which are listed in Table 5.2. The terrain is gravel on hard ground, and its parameters are also listed in Table 5.2. Since we cannot find the parameters of the gravel on hard ground from any source, we decided to use the parameters of dry sand, which closely represent the same terrain characteristic. In the experiment, the MBC was driven to make a 180-degree turn twice. The motions of the MBC were recorded by using a camcorder, and the radii of turning were measured. After the recorded tape was analyzed, the experimental results from two experiments were averaged. The results obtained from the experiments are shown below:

$$v_x = 1.12 \quad \text{ft/s}$$

ω = 0.10 rad/s

iR = 0.145

iL = 0.15

Rt = 11.04 ft

The elapsed time that the MBC takes to make a 180-degree turn is 31 seconds. When we compare the results from the simulation with that from the experiment, it clearly shows that the model can predict the real event closely. Thus, we can accept at one level that the model is reliable. However, to ensure the accuracy and fidelity of the model, more extensive experiments are needed either on a full-scaled MBC or a scaled one.

5.4 The simulation of the CHS

The program for simulating the CHS is a modified version of the simulation program for single MBC. Many functions are added into it; however, its main part related to tracked vehicle dynamics is maintained. The flowchart diagram and source codes are shown in Appendix B. The program has the capability to simulate the CHS with any number of MBCs in the system by simply making some changes to the main program. In this section the simulation of a two-unit CHS will be presented. However, the model validation in this case is difficult and costly. Therefore, the validation step will be neglected.

The main inputs to the program comprise MBCs' data, Dollys' data, Pigs' data, and terrain parameters. The outputs are separated into two parts. One is the graphic animation display, and the other is graph plotting. The user can plot a variable of interest, which is chosen from a wide range of variables such as each MBC's variables, Dollys' reaction forces, and connecting pins' forces, by making small changes in specified areas in the main program. The following is the simulation of a two-unit CHS. In this simulation, both MBCs are parked along side of the RFM, and their heading direction pointing to the right. The leading MBC is driven straight while the following MBC is driven to turn in a clockwise direction with a zero-radius of turning. The simulation keeps running until the following MBC's Dolly reaches the travel limit, and the leading MBC starts to drag the following one for a second before the simulation

stops. Figure 5.9 shows a snap shot from a computer monitor while the simulation is running.

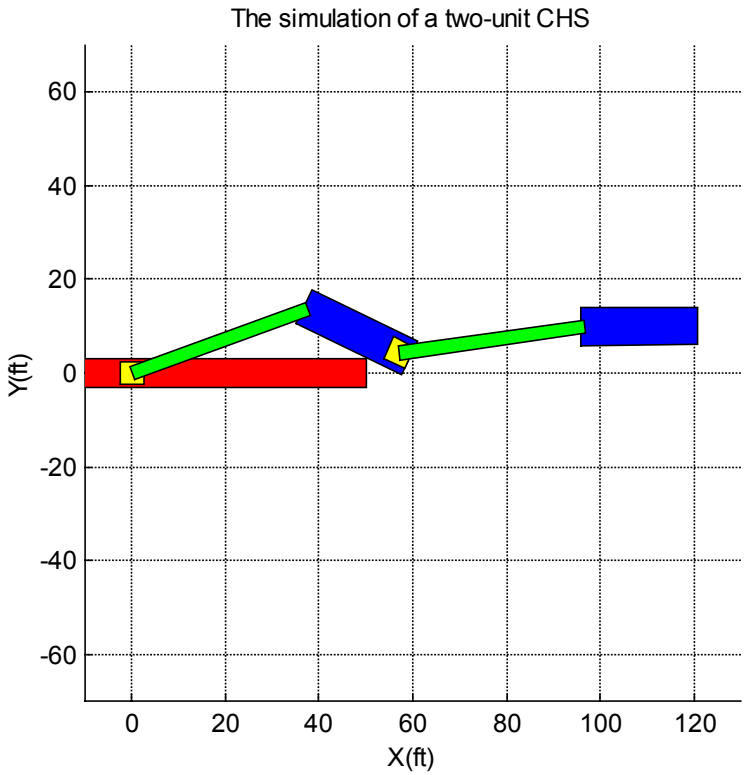

Figure 5.9. A Snap Shot from the Simulation of a two-unit CHS

1st MBC	
Vr	1 ft/s
Vl	1 ft/s
2nd MBC	
Vr	-1 ft/s
Vl	1 ft/s
Pig	
W_P	25000 lbf
W,pig	3 ft
L,pig	40 ft
L	40 ft
Dolly	
W_D	1000 lbf

Table 5.3. Input Parameters

The inputs to the program are the same as shown in Table 5.2 for MBCs' and terrain parameters. Only the velocity commands for each MBC are changed, and both Dollys' and Pigs' parameters are added into the simulation program as shown in Table 5.3. From Fig. 5.9, the M**B**Cs, Doll**y**s, Pi**g**s, and **R**FM are represented by blue, yellow, green, and red rectangles, respectively. The following figures are graphs plotted at the end of the simulation showing the velocity profiles of both MBCs, connecting pin forces, normal force, and torque over the simulation time, 3 seconds.

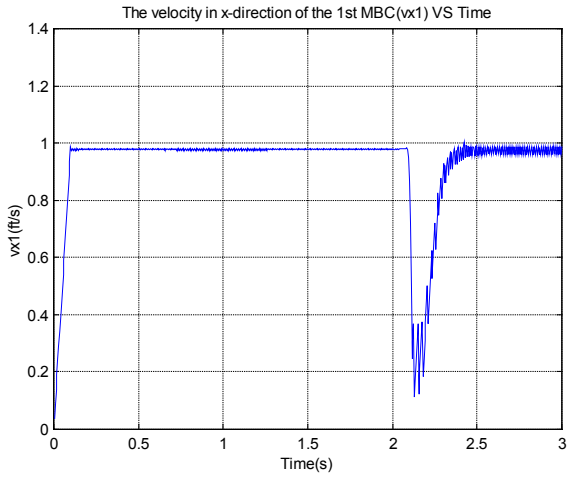

Figure 5.10. The 1st MBC's v_x VS Time

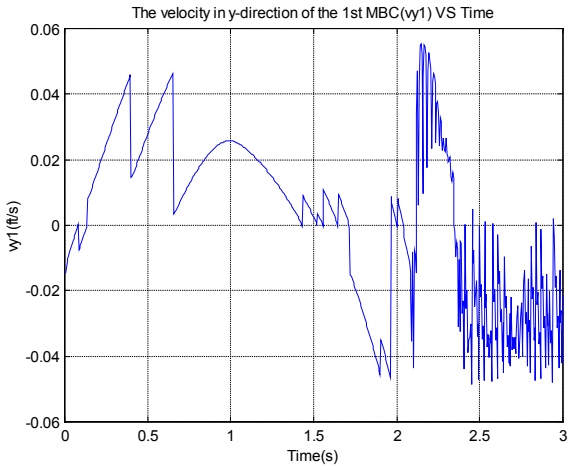

Figure 5.11. The 1st MBC's v_y VS Time

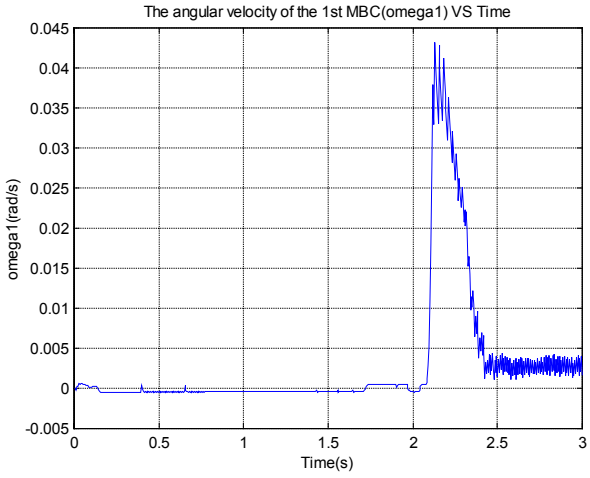

Figure 5.12. The 1st MBC's ω VS Time

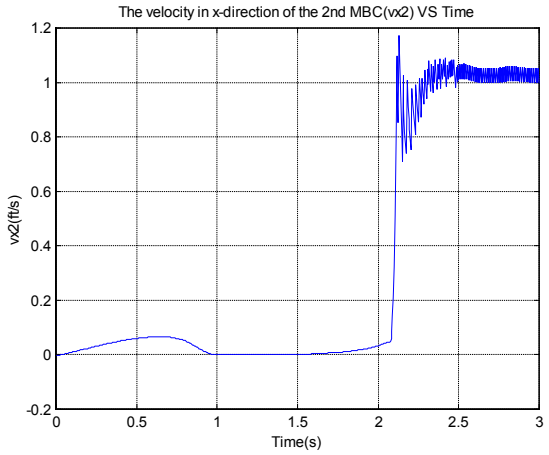

Figure 5.13. The 2nd MBC's v_x VS Time

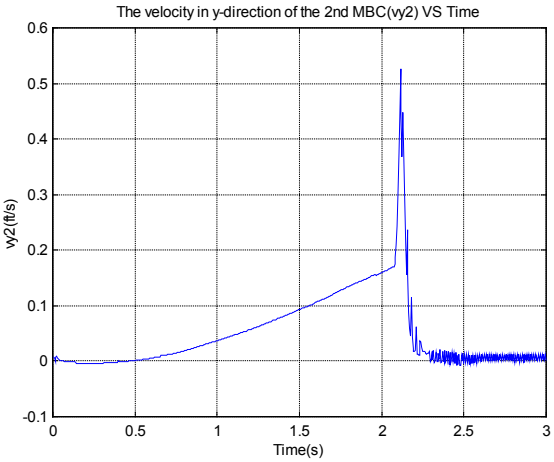

Figure 5.14. The 2nd MBC's v_y VS Time

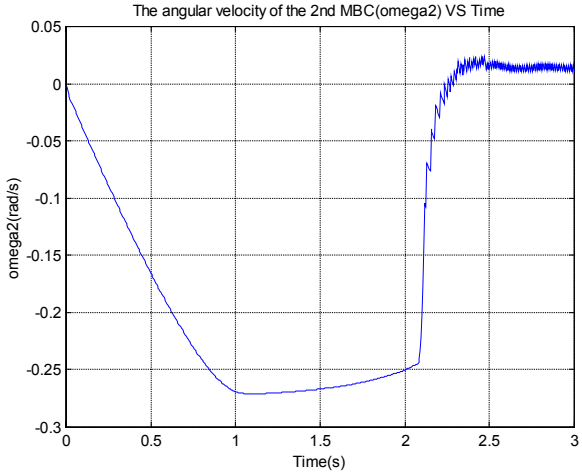

Figure 5.15. The 2nd MBC's ω VS Time

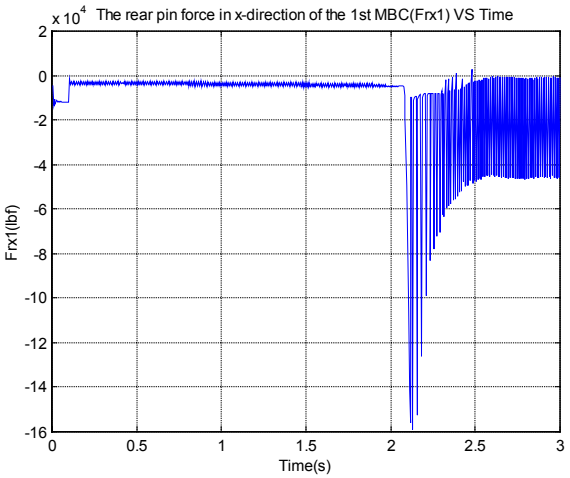

Figure 5.16. Frx1 VS Time

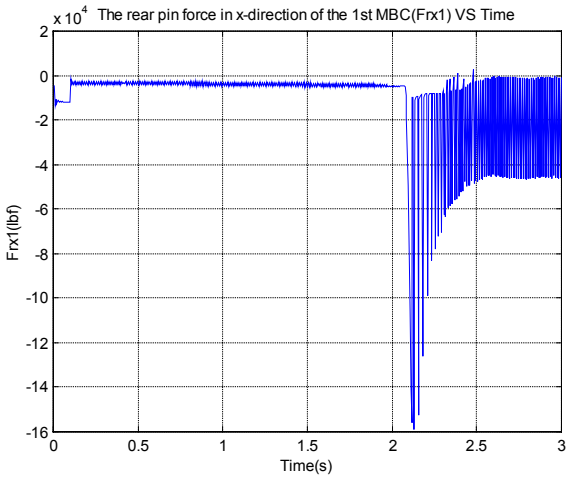

Figure 5.17. Fry1 VS Time

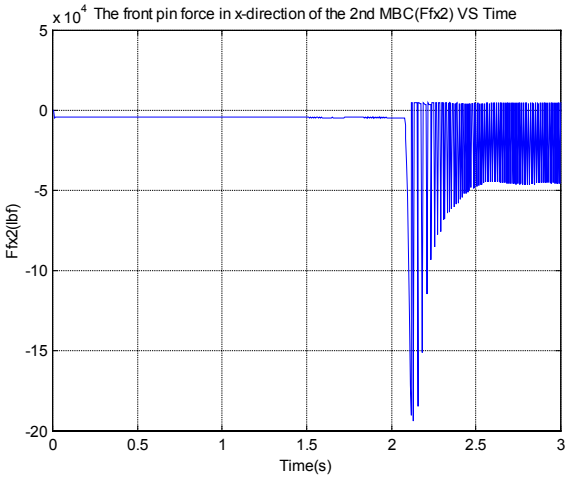

Figure 5.18. Ffx2 VS Time

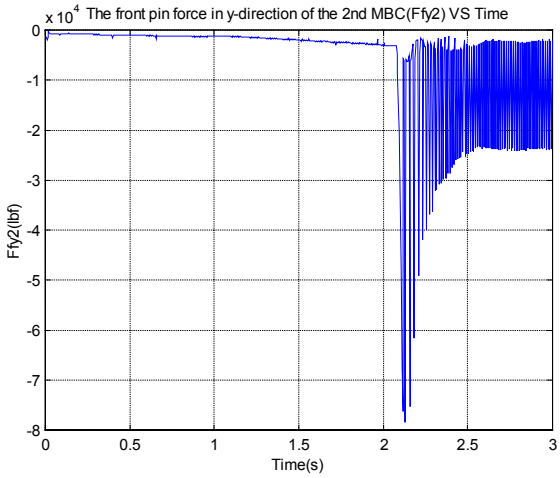

Figure 5.19. Ffy2 VS Time

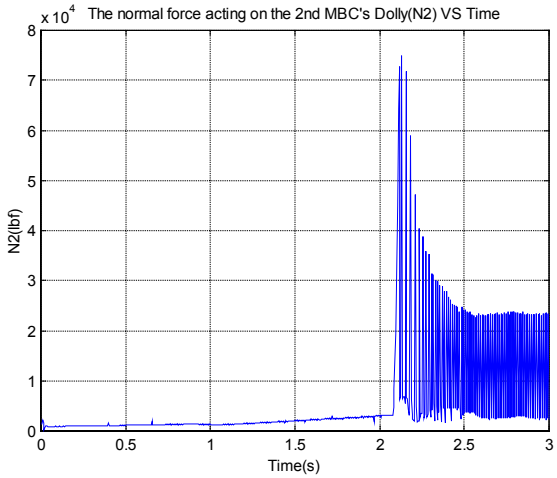

Figure 5.20. Normal Force VS Time

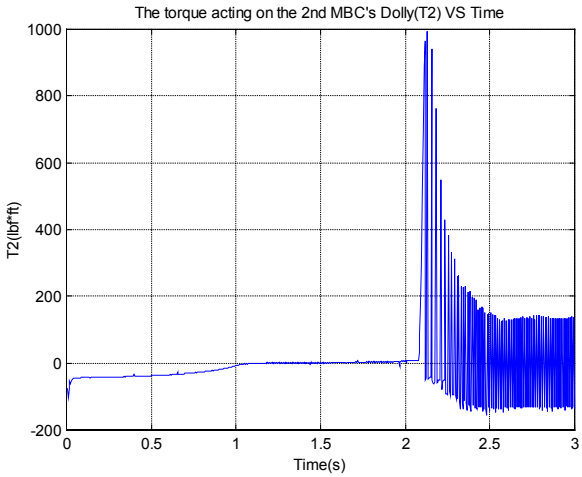

Figure 5.21. Torque VS Time

From these graphs, it is obvious that the Dolly reaches its travel limit at time 2.1 seconds after the simulation began by observing that the values of MBCs' velocities and forces start to fluctuate. The information from these graphs can be used in many applications such as in designing the size of the connecting pin and donut. Moreover, the program can be applied to be used as a test bed for other purposes. For example, the early stage of the development of the line finding algorithm using a laser scanner is developed and tested on the simulation before it is implemented on a one-tenth-scale model. The snap shot from a computer monitor of this simulation is shown in Fig. 5.22.

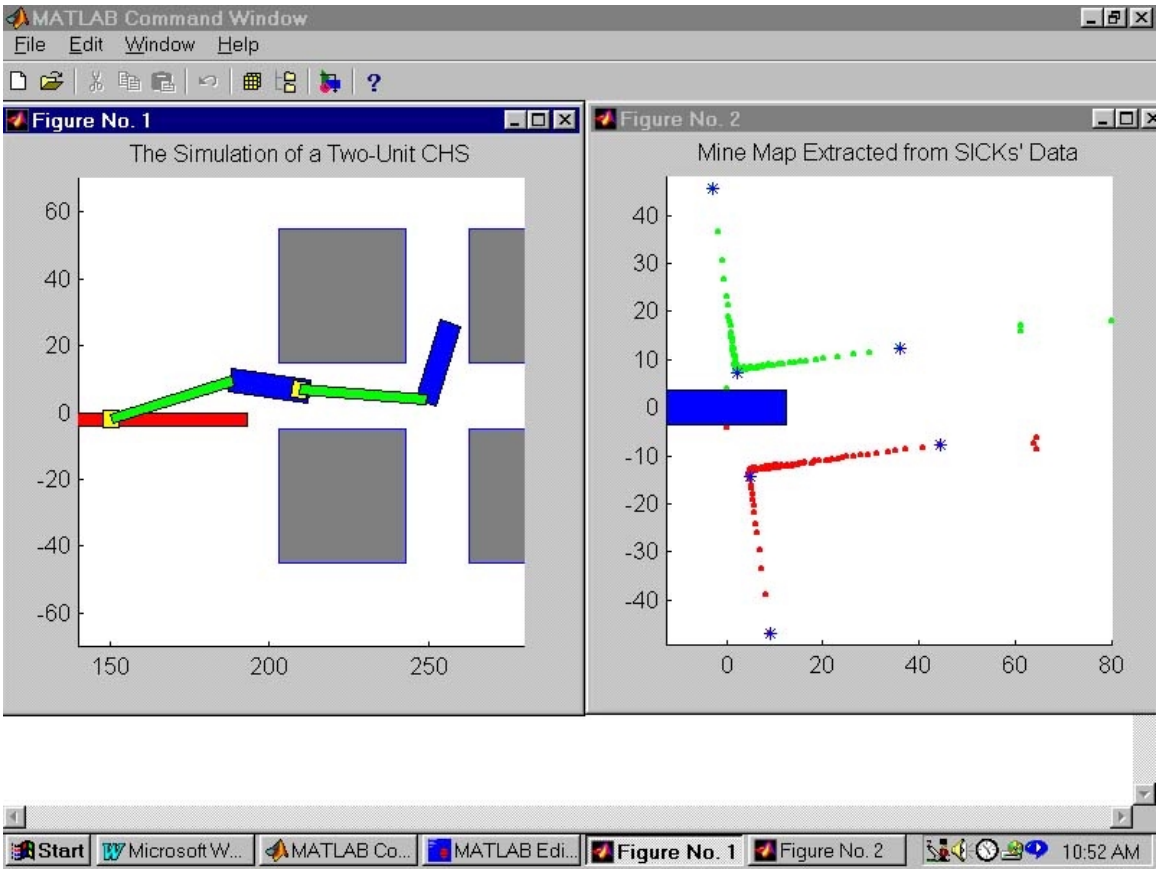

Figure 5.22. A Snap Shot of the Simulation of the SICK's Line Finding Algorithm

From Fig. 5.22, the program is simulating a two-unit CHS being operated in a room-pillar type mine as shown in the small window on the left. Both MBCs are manually driven, and two laser scanners are installed on the following MBC. The scanned range data obtained from the laser scanners are plotted in the small window on the right, and distinguished by red and green colors for the right and left laser scanners, respectively. The blue asterisks added to the plot indicate the locations of corners of mine pillars extracted from the scanned range data by the line finding algorithm.